Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 89 (2016) 162 – 169
Procedia Computer Science

Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016)

A Distributed Minimum Spanning Tree for Cognitive Radio Networks

Mahendra Kumar Murmu, Akheel M. Firoz, Sandeep Meena, and Shubham Jain*

National Institute of Technology, Kurukshetra, Haryana 136 119, India

Abstract

The minimum spanning tree is a classical problem in distributed system environment. We extend this challenge in Cognitive Radio Networks (CRN). In CRN, the spectrum mobility and the node mobility creates connectivity problem during neighbour discovery. Thus, finding edges (or relation graph) between the SU nodes in order to create communication graph for Minimum Spanning Tree (MST) is a challenge in cognitive radio network. In the present work, we propose a solution to the problem of creating minimum spanning tree (MST) in cognitive radio network. It is a message passing based distributed algorithm. The MST algorithm find shortest path between any pairs of SUs (or vertices) in the communication graph of CRN. The communication message complexity of our algorithm is $6E$, where E represents the edges. The MST is useful for data dissemination in cognitive radio network.

Peer-review under responsibility of organizing committee of the Organizing Committee of IMCIP-2016

Keywords: Cognitive Radio Network; Minimum Spanning Tree; Minimum Cost; Distributed Algorithm.

1. Introduction

The Cognitive Radio Networks (CRN) have paid great attention for efficient spectrum utilization in wireless communication environment, and popularity increases more. The CRN fulfils the gap of spectrum scarcity problem observed in[1]. In a typical cognitive radio scenario, users of a given frequency band are classified into Primary Users (PU) and secondary users (SU). The primary users are licensed users of that frequency band. The secondary users, on the other hand, are unlicensed users that opportunistically access the spectrum when no primary users operating on that frequency band. The networks formed in this category are sometimes called cognitive radio ad hoc networks (CRAHN) and details can be seen in[2].

In CRN, the autonomous SU nodes are equipped with the following characteristics: learning, efficiency, intelligence, reliability, and adaptivity[10]. On the other hand, the channels have the numerous properties such as spatial variation, spectrum fragmentation, and temporal variation[8]. The communication channels to the SU nodes are available as long as the channels are free from the PU interference in the radio environment. The channel sensed by each SU node is termed as the Local Channel Set (*LCS*). From *LCS*, some channels are used to connect SU nodes, logically, are called common control channel (*CCC*). Therefore, *CCC* belongs to the *LCS* (i.e., $CCC \in LCS$). However, at any time only one *CCC* is used to represent logical relation between any two neighbors. Therefore, *CCC* forms the edges

*Corresponding author. Tel.: +91-89-501-18-619.
E-mail address: mkmurmunitkkr@gmail.com

Peer-review under responsibility of organizing committee of the Organizing Committee of IMCIP-2016
doi:10.1016/j.procs.2016.06.030

among the nodes. The total number of channels sensed by all SUs are connected in the communication graph is referred to as Global Channel Set (*GCS*). Hence, the *GCS* is called as superset of *CCC*, where *CCC* and *LCS* both belong to *GCS*. The neighbor discovery problem is not only depends on CCC but also on distance and the radio range[14]. Thus, obtaining set of edges in order to create minimum spanning tree graph is a challenge in cognitive radio networks. The minimum spanning tree is useful for many applications such as data dissemination, routing, coordinator election etc.

The minimum spanning tree graph is a collection of logically connected sub tree graph. The high link variation is observed between any neighbors due to the PU appearance for its channel. The edge (or logical link) created by *CCC* between neighbors carries some cost (or weight). The associated cost may vary as per the location of SU nodes in the network. In other side, the same cost may be associated with one or more *CCC* if the SU node does not change its location. Therefore, the complexity of the network connectivity increases due to the spectrum and SU node mobility in cognitive radio networks[8]. In CRN, it is always a difficult task to create minimum spanning tree and not easy to declare the termination. In this paper, our objective is to connect minimum weighted edges to construct MST that connects all SU nodes (or vertices) in the communication graph. The resultant tree is an acyclic graph in cognitive radio network. The more reliable and preferable situation for connecting sub graph of SU nodes is if the degree of *CCC* is more. If such a condition occurs, then the priority is always given to the lowest edges (or sub graph) and highest connectivity channel for MST in CRN.

The construction of minimum spanning tree is a fundamental problem in distributed computing system. The various authors in[3,4,7,11–13] have proposed distributed algorithms to construct MST in wireless networks. During MST construction, routing is an important aspect in the network. In our problem, we have taken the idea of works in[9] that describes details of reactive routing which helps us to route the packet or message delivery. In CRN, MST construction is an inherent challenge. Some application oriented MST for cognitive radio networks has been created by numerous authors in[5,10] such as for efficient multicasting and termination detection.

The present work proposes decentralize distributed algorithm that constructs a minimum spanning tree for cognitive radio network. All the nodes have equal priority to calculate the MST in CRN. The actions of SUs are distributed and it is local and separable in cognitive radio network. The MST grows from both ends of the nodes connected by an edge. However, when the algorithm terminates, the resultant tree at the end of computation is unique. In the algorithm, each secondary user node is uniquely identified by the identifier (*id*). Each SU node maintains a local channel set. The *id* and the *LCS* are to be transmitted among the nodes using message. From *LCS*, at least one from *CCC* set is used to create edges among the SU nodes that carry some cost. At each step, sub graph information is broadcasted to all neighbors except from where the message has been received. The receiver node send acknowledgement to the sender. Once, all nodes are acknowledged by its neighbors, the resultant tree is termed MST of the CRN. The nature of our algorithm is pure distributive.

The rest of the organization of the paper is as follows. The section 2 describes the works related of MST construction in asynchronous model. Then we propose a system model for MST in cognitive radio network in section 3. The section 4 describes a state diagram of our proposed work. The detail description of the proposed algorithm is explained with the help of flow chart in section 5. The section 6 measure the complexity of our algorithm. Then we also verify our algorithm by measuring correctness proof in section 7 and finally conclusion is given in section 8.

2. Related Work

The minimum spanning tree has been well explored in the distributed system environment. The message efficient MST algorithms have been proposed in[3,4]. The work of[3] describe distributed algorithm in terms of multi processor interaction where the mobility of nodes are exempted in the network. The weights of the edges are considered distinct. The other[4] described the distributed environment in terms mobility of the nodes has been added whereas the edge weights have been considered as un varied. In ref. [5, 10] proposed for some specific application oriented to meet quality of service (QoS) requirements to be achieved in the cognitive radio networks. The prior[5] is for multicasting and[6] is for termination detection. Some other distributed MST algorithms have been proposed in[7,11–13]. In this paper, our approach is to design a distributed algorithm that construct minimum spanning tree in cognitive radio networks. It is a decentralized based approach.

3. System Model

Let's assume a cognitive radio network consists of n SU nodes and k channels. In CRN, the communication weighted graph $G = (V^*, E^*, LCS)$ represents the vertices (V_i) are connected with the help channels, later termed as edges (E_i). The SU nodes are called the vertices and common control channels between nodes (n_i, n_j) is termed as edge. There is a cost associated with the each edge. This may be a transmission cost between any nodes. The local connected nodes show the relation graph among them and it is denoted as fragment. There may exist more than one *CCC* between any SU nodes. However, at any time only one *CCC* is associated among the nodes in MST. Similarly, there may exist more than one links between any two SU nodes, but, eventually there will be exists only one unique path from any source to destination nodes and the resultant combination of fragment is termed as minimum spanning tree of the cognitive radio network. The tree is acyclic and unique that connects all SU nodes with edges and the total edges are one less than the nodes are connected. In this algorithm, we have not considered the activity of PU.

We have used four different states of nodes and edges. The descriptions are as following.

3.1 States of nodes

Sleeping: The node simply waits for external instructions or just in sleep state.
Find: Prepared LCS list, send participation message.
Found: The request made by the sender has been confirmed by the receiver and fragment is build.
Lock: The state ensures the partial knowledge of shortest path in order to build MST in CRN.

3.2 States of edges

Baisc: The edge has not been marked for any specific purpose.
Rejected: The edges which is currently not used for transmission is consider as rejected edges.
Temporary Branch: If an edge is the best possible alternative to the best transmission edge possible, then it is added to the temporary branch set.
Branch optimum: The confirmed optimum edges.

3.3 Data structure used

id: The connected n nodes are uniquely represented in the network.
LCS: The channel set sensed by each SU node, locally.
CCC: List of common channels with the neighbors.
GCS: Total k channels sensed by the SU nodes in CRN.

3.4 Message types

beacon_request(*id*, *LCS*): The message is used for connection request.
ack_reply(*id*, *LCS*): The message is used to grant the request message.
update_CCC(*id*, *CCC*): The message is used to grow the network topology albeit temporarily.
minimal_tree_topology(*id*, *CCC*): The message is used to inform neighbor nodes about local minimal tree.

4. State Diagram

The state diagram shown in Fig. 1 represents the change of states of the running process of node and channel during findings of MST graph in cognitive radio networks. Initially, the SU nodes are in *Sleeping* state. The nodes in this state are not the part of any cognitive radio ad hoc networks. The state of the nodes changes when the external input is received or the node waked up by itself. Once awakened, the node (or process) changed its state into *Find* state. In this state, the nodes are self aware about its *LCS* and waits for acknowledgement from other. Initially, the channel or edge remains in the *Basic* state. The node generates beacon request in each channel for neighbor discovery. On receiving,

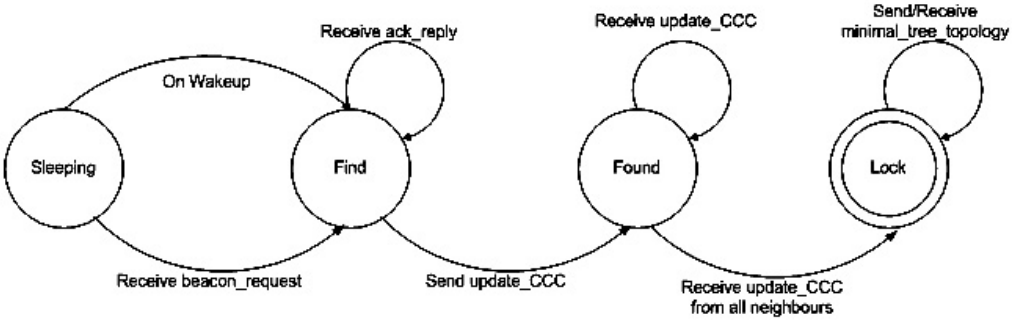

Fig. 1. State diagram representation of the Algorithm.

the receiver sends reply message using *ack_reply* to the sender. Once acknowledgement is received, the node turns its state into the *Found* state. The state of the selected path from any source to destination is reached into the *Temporary* Branch state. The node starts comparison to find the optimum between the neighbors. If there are multiple paths connected with *CCC*, then best one is used for connection and rest is stored in the *Rejected* state for future references. The list of temporary edges may not necessarily be the part of MST in the final stage. After completion of the each local computation, the state of the nodes reaches into the *Lock* state and the state of the edges becomes *Branch Optimum* state. The local optimum path is selected among each participating node for MST in CRN. The fragment grows and when all local *Branch Optimum* is collected the resultant tree is called the minimum spanning tree in cognitive radio networks.

5. Flow Chart of the Algorithm with Description

5.1 Algorithm description using flow chart

We describe our proposed algorithm using flow chart to obtain a MST in CRN. We use node states and edge states in order to identify the stages of the running algorithm. The algorithm is based on message passing mechanism. Initially, the nodes are in the *Sleeping* state depicted in the flow chart of the algorithm Fig. 2. In algorithm it is shown in step 1 in Fig. 3. In this state, the SU node waits for the external instruction in order to initiate MST construction in cognitive radio network. The node wake up or on receiving *beacon_request*, the node change its state into *Find* state and initiates MST construction. The flow of the process in given in Fig. 2 and in algorithm it is given in step 2. In the *Find* state, the SU node starts sensing to find *LCS* and broadcast *beacon_request* message to all neighbors. The node waits for the acknowledgement from others. The edges of the nodes remain in the *Basic* states. The message includes *id* of the sender node and the *LCS*. On the other hand when the SU node receives *beacon_request* message, they immediately generates an acknowledgment to neighbor from where the request has been received. On receiving *ack_reply*, the state of the node turns into the *Found* state and starts comparison for destination node i in both of the *LCS*. The path containing optimum value towards destination node is associated to *CCC* and hence edge is considered as branch. We also maintain table for other alterative optimum path. If available, the other alternative path towards destination is saved into the *Temporary Branch* state otherwise it is considered in a *Rejected* state. It is given in our algorithm in step 5 in Fig. 3 also depicted in the flow chart. The edge selection may be based on the basis of signal strength or some other parameters which represent weighted cost of the channel. The saved channel is useful as a fallback in case of failure of the *Branch Optimal* edge, in order to ensure QoS of the CRN. We select always a lower cost edges and placed higher cost edges into the *Rejected* state. The overall process is termed as fragment creation. During MST creation, a node is aware of its local fragment. To grow the MST, we use *update_CCC* message that carries node*id* and *CCC* in the network. The node broadcasts *update_CCC* to all neighbors. The state of the node remains in *Found* state. On receiving *update_CCC*, the node starts comparison for destination node i in both of the *CCC*. We select best possible path from the obtained channel set and makes it a part MST. The process steps are given in step 7 of our algorithm in Fig. 3 and also depicted in the flow chart. After updating optimal *CCC*, the node forward MST graph

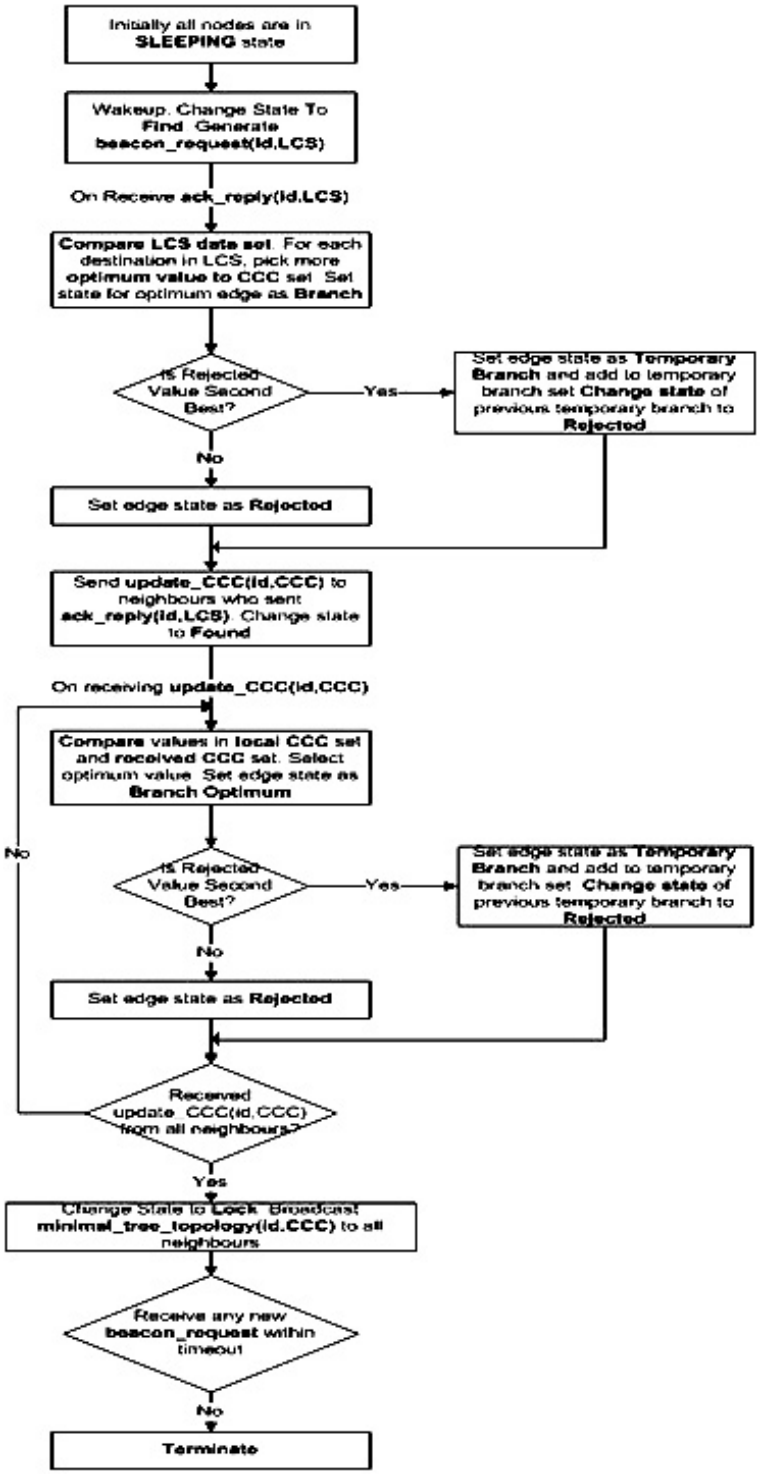

Fig. 2. Flow chart of the Algorithm.

information using *minimum_tree_topology* to all neighbors. A *minimum_tree_topology* message is use to share all the knowledge MST to the neighbors. The step increases the more accurate results towards getting MST graph in cognitive radio network. Once receiving of *minimum_tree_topology* from neighbors, the node turns into *Lock* state and the state of the channel reached to the *Branch optimum* state. When no more *beacon_request* is received within timeout, the MST algorithm is terminated where all SUs are connected with exactly $n - 1$ edges and the resultant tree becomes unique and acyclic in cognitive radio network.

5.2 Proposed algorithm

1. Initially, all nodes are in *Sleeping* state.
2. On wake up, the node reaches into the *Find* state and sensed *LCS*. The node generate *beacon_request*(*id*, *LCS*) to its neighbours and the edges is considered into *Basic* state.
3. On receiving *beacon_request*(*id*, *LCS*), the receiver generate *ack_reply*(*id*, *LCS*).
4. If the node receives beacons as in step 3 and node remains in sleeping state
   then generate *beacon_request*(*id*, *LCS*) and send to neighbors and changes state to *Find*.
5. On receiving *ack_reply*(id, *LCS*), start comparison between the sender and receiver *LCS*.
   If destination node '*i*' is present in only one of the two *LCS*'s
   then the *CCC* takes from that *LCS*.
   Else if destination node '*i*' is present in both *LCS*
   then take minimum cost and add to *CCC*.
   Second best *LCS* is saved in *Temporary Branch* state. Rejected sets from *LCS* (both) are changed to *Rejected* state. Node switches to *Found* state.
6. Send *update_CCC*(*id*, *CCC*) to all neighbours and the node state remains in the *Found* state.
7. On receiving *update_CCC*(*id*, *CCC*), start comparison between sender and the receiver CCC.
   If destination node '*i*' is present in only one of the two *CCC*'s
   then the optimal *CCC* takes the cost of transmission from that *CCC*.
   Else if destination node '*i*' is present in both *CCC*
   then take minimum cost and add to optimal *CCC*.
   Second best *CCC* is saved in *Temporary Branch* State. Rejected sets from *CCC* (both) are changed to *Rejected* state. Node remains in the *Found* state and the edges become *Branch optimum*.
8. After updating *optimal CCC*, i.e. received *update_CCC* from all neighbours, the node forward *minimal_tree_toplogy*(*id*, *CCC*) message to all neighbours.
9. On receiving *minimal_tree_topolgy*(*id*, *CCC*) from all neighbours, the state of the node reached into *Lock* state and edge become *Branch optimum* and algorithm is terminates.

Fig. 3. Distributed MST Algorithm.

6. Message Complexity of the Algorithm

In this algorithm, initially a node sends at least one *beacon_request* message to its immediate neighbor and then receives an *ack_reply* message for the same. This indicates the transfer of two messages. On receiving the *ack_reply* from the neighbor, the sender node generates own channel set and sends an *update_CCC* message. Similarly, the receiver node sends an update back to the sender node, which will cause it to update its own channel set. Therefore, there is two *update_CCC* message passed. In the same order like *update_CCC*, the *minimum_tree_topology* is propagated twice in the edges between the nodes. Therefore, edges require 6 messages to be passed. Hence, the message complexity of our algorithm is $6E$, where E indicates the number of edges.

7. Correctness Proof of the Algorithm

We assume n nodes and k channel passes through different and distinct process states during the execution of the algorithm. Any two vertices (V_i) are associated with one weighted edge (E_i) in the MST. Here, the vertex

represents the secondary user node, edges is the *CCC*, where $CCC \in LCS$. The nature of the MST algorithm is progressively processes from both end of the node connected by an edge. The resultant tree is computed as MST $(G) = (V^*, E^*, LCS)$, which is unique and acyclic graph in cognitive radio network.

Lemma 1. There is no cycle in the graph.

Proof: Let us assume that a cycle exists in the *MST* in cognitive radio network.

For a cycle to exist there must be a path from a node back to itself without visiting any edge twice. But, as per the flow of the algorithm is created, there exists only one path from one node to another and it passes on the message to the next node indicated in its *CCC*. A message could return back to the initial node through unique paths provided in the initial input network graph. But, those edges after execution of the algorithm are converted to *Branch Optimal*, *Temporary* Branch or *Rejected* state. Therefore, a message cannot make it back to the initial node unless it uses a *Rejected* or *Temporary* branch edge, which is not permitted for the messages after the algorithm has terminated. Thus, by contradiction, we prove that no cycle exists in the algorithm.

Lemma 2. When the algorithm terminates, $n - 1$ edges cover n vertices and resultant tree is *MST*.

Proof: Let us assume that for the MST to be valid, the number of edges must be equal to the number of nodes in the network, i.e. $n - 1$ edges implies $n - 1$ nodes exist in the network.

From the definition of MST, we have observed that $n - 1$ edges covers path between n nodes. The flow of our algorithm results in a MST, and unique paths exist between nodes. Thus, we cannot have equal number of nodes and edges unless a cycle exists in the MST of the network but, in Lemma 1 we have proven that as a result of execution of our algorithm, there exist no cycle in the network. Thus, by contradiction, we prove that there exist $n - 1$ edges for n nodes in the network.

8. Conclusions

We proposed distributed algorithm based on message passing mechanism that construct a minimum spanning tree in cognitive radio network. The complexity of our algorithm is $6E$, where E is the edges of the MST algorithm in cognitive radio network. The correctness proof of the algorithm is included. We have successfully conducted an implementation of our proposed work in C++. In this algorithm, we have not included the presence of PU. Our planning is to extend the algorithm by involving behavioral activity of PU and measure the performance of MST construction under different failure condition. We are also looking for simulating our results in NS-2. The proposed minimum spanning tree is simple and distributive for cognitive radio network.

References

[1] I. F. Akyildiz, W. Y. Lee, M. C. Varun and S. Mohanty, NeXt Generation/Dynamic Spectrum Access/Cognitive Radio Wireless Networks: A Survey, *Computer Networks*, vol. 50, pp. 2127–2159, 15 September (2006).
[2] I. F. Akyildiz, W. Y. Lee and K. R. Chowdhury, CRAHNs: Cognitive Radio Ad Hoc Networks, *Ad Hoc Networks*, vol. 7, pp. 810–836, July (2009).
[3] R. G. Gallager, P. A. Humblet and P. M. Spira, A Distributed Algorithm for Minimum-Weight Spanning Trees, *ACM Transaction on Programming Language and Systems*, vol. 5, pp. 66–77, (1983).
[4] I. Lavallee and G. Roucairol, A Fully Distributed (Minimal) Spanning Tree Algorithm, *Information Processing Letter*, vol. 23, pp. 55–62, (1986).
[5] L. Xie, X. Jia and K. Zhou, QoS Multicast Routing in Cognitive Radio Ad Hoc Networks, *Int. J. Commun. Syst.*, vol. 25, pp. 30–46, (2012).
[6] H. M. Almasaeid, T. H. Jawadwala and A. E. Kamal, On-Demand Multicast Routing in Cognitive Radio Mesh Networks, *GLOBECOME*, pp. 1–5, (2010).
[7] M. Khan and G. Pandurangan, A Fast Distributed Approximation Algorithm for Minimum Spanning Trees, *Distrib. Comput.*, vol. 20, pp. 391–402, (2008).
[8] H. Liang, T. Lou, H. Tan, Y. Wang and D. Yu, On the Complexity of Connectivity in Cognitive Radio Networks through Spectrum Assignment, *J. Comb Optim.*, vol. 29, pp. 472–487, (2015).
[9] A. S. Cacciapuoti, M. Caleffi and L. Paura, Reactive Routing for Mobile Ad Hoc Networks, *Ad Hoc Networks*, vol. 10, pp. 803–815, (2012).

[10] S. Sharma and A. K. Singh, On Termination Detection in Cognitive Radio Networks, *Network Management*, vol. 24, pp. 499–527, November (2014).
[11] A. K. Singh, A. Negi, M. Kumar, V. Rathee and B. Sharma, On the Linear Time Construction of Minimum Spanning Tree, *Int. Conf. on Recent Trends in Information, Telecommunication and Computing*, pp. 1–6, (2014).
[12] Z. Lotker, B. Patt-Shamir, E. Pavlov and D. Peleg, Minimum-Weight Spanning Tree Construction in $O(\log \log n)$ Communication Rounds, *SIAM J. COMPUT.*, vol. 35, pp. 120–131, (2005).
[13] B. Awerbuch, Optimal Distributed Algorithms for Minimum Weight Spanning Tree, Counting, Leader Election and Related Problems, *ACM Symposium on Theory of Computing*, pp. 230–240, (1987).
[14] A. A. Khan, M. H. Rehmani and Y. Saleem, Neighbor Discovery in Traditional Wireless Networks and Cognitive Radio Networks: Basics, Taxonomy, Challenges and Future Research Directions, *J. of Computer Applications*, vol. 52, pp. 173–190, (2015).